WHAT IS CLAIMED IS:

1. A method of operating a plurality of business software components, the method comprising:

discovering information about a first business software component, the first business software component having at least one first component capability or first component requirement;

binding the at least one first component capability or first component requirement to a first role in a model-driven bus;

discovering information about a second business software component, the second business software component having at least one second component capability or second component requirement; and

binding the at least one second component capability or second component requirement to a second role in a model-driven bus.

2. The method of claim 1, wherein the step of discovering information about the first business component is performed by a discovery manager.

3. The method of claim 2, wherein the step of discovering information about the second business component is performed by a discovery manager.

4. The method of claim 1, wherein the step of discovering information about the first business component occurs automatically.

5. The method of claim 4, wherein the step of discovering information about the second business component occurs automatically.

6. The method of claim 4, wherein the automatic discovery occurs as part of installation of the first business software component.

7. The method of claim 1, wherein the information about the first business component is metadata.

8. The method of claim 7, wherein the information about the second business component is metadata.

9. The method of claim 1, wherein at least one capability of the first business software component overlaps at least one capability of the second business software component, and wherein the model-driven bus provides arbitration such that only one of the first and second business software components provides the overlapping function.

10. The method of claim 1, and further comprising:

providing standardized messaging between the first and second business software components.

11. The method of claim 1, and further comprising:
examining role bindings to determine if a business process can be enabled.

12. The method of claim 11, wherein examining includes comparing process role bindings to predefined process pattern information.

13. The method of claim 12, wherein the predefined process pattern information is part of a pattern fitness layer.

14. A business software system comprising:
a software bus having a temporally stable interface designed in accordance with a comprehensive business taxonomy;
a first business software component bound to and fulfilling a first portion of the software bus; and
a second business software component bound to and fulfilling a second portion of the software bus.

15. The system of claim 14, wherein the software bus includes a message routing layer for communication with each of the software components.

16. The system of claim 14, wherein the software bus includes a pattern fitness layer to check information relative to the first and second software components.

17. The system of claim 14, wherein the software bus includes an administration layer to facilitate user management of the components.

18. The system of claim 14, wherein the software bus includes a replication layer.

19. The system of claim 14, wherein the software bus includes an auditing layer.

20. The system of claim 14, wherein the software bus includes a key performance indicators layer.

21. The system of claim 14, wherein the software bus is usable with different comprehensive business taxonomies.

22. The system of claim 21, wherein each of the different comprehensive business taxonomies is domain-specific.

23. A computer readable medium having instructions stored thereon, the instructions defining a standardized adapter comprising:

a software component side custom-configured to interact with a specific business software component; and

a standardized side coupled to the software component side and adapted to interact with a standardized, durable application

programming interface, wherein the standardized side includes data relative to at least one business process that is not supported by the software component.

24. A method of retrofitting a stand-alone business software component for use in an automatically integrating business software system, the method comprising:

describing capabilities of the stand-alone business software component with metadata;

describing requirements of the stand-alone business software component with metadata; and

generating a standardized software adapter to interface the stand-alone business software component to the integrated business software system.